Adding Boolean-quality control to best-match searching via an improved user interface

Donald Byrd and Rodion Podorozhny

Center for Intelligent Information Retrieval
Computer Science Department
University of Massachusetts
Amherst, MA 01003
{ dbyrd , podorozh }@cs.umass.edu

1 May 2000

Abstract

While end users these days seem happy with best-match text-retrieval systems, it appears that expert searchers still prefer exact-match (Boolean) text-retrieval systems by an overwhelming margin. This is somewhat surprising. Most expert searchers were probably trained with Boolean systems, and an obvious factor is simply preferring the familiar, but we argue that a second major factor is that these experts feel a much greater sense of control with Boolean than with best-match systems. We have designed a best-match system, MIRV, incorporating user-interface features that we believe will give experts a sense of control comparable to that of Boolean systems and that we believe end users will also be happy with. We implemented MIRV's document viewer and did a controlled user study, with encouraging results.

This material is based on work supported in part by the National Science Foundation, Library of Congress and Department of Commerce under cooperative agreement number EEC-9209623, and also supported in part by Defense Advanced Research Projects Agency/ITO under ARPA order number D468, issued by ESC/AXS contract number F19628-95-C-0235.

Any opinions, findings and conclusions or recommendations expressed in this material are the author(s)' and do not necessarily reflect those of the sponsor(s).

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **01 MAY 2000**	2. REPORT TYPE	3. DATES COVERED **-**

4. TITLE AND SUBTITLE
Adding Boolean-quality control to best-match searching via an improved user interface

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Defense Advanced Research projects Agency,3701 North Fairfax Drive,Arlington,VA,22203-1714

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT
see report

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE		**30**	
unclassified	**unclassified**	**unclassified**			

1. Rationale

The advent of the World Wide Web has resulted in an explosion of searching by end users as opposed to expert intermediaries. The vast majority of end users are not expert searchers, and it seems clear that they generally prefer to enter their queries in free-text (very often but very misleadingly called "natural language") form rather than in any type of structured input, whether with Boolean operators or otherwise. Free-text input generally implies best-match rather than exact-match evaluation.

This approach fits well with recent experiments (Turtle 1994), which confirm indications of many years earlier (Sparck Jones 1981) that free-text-input best-match text-retrieval systems get results in nearly all cases at least as good as, and often much better than, exact-match (Boolean) systems. Hersh and Hickam (1995) survey prior research comparing Boolean and best-match searching; results are not completely consistent, but most of the small number of studies support Turtle. Furthermore, there are substantial theoretical grounds for expecting such a result. For example, Blair and Maron (1985) argue strongly that no conceivable full-text retrieval system can achieve consistently good recall with a large database, but their arguments apply only to exact-match evaluation: there is no theoretical reason why best-match systems cannot produce good recall with databases of any size. Finally, Sparck Jones and Willett (1997, p. 3) comment "The Boolean model…is entirely appropriate for use in a DBMS context…However, it is far less obviously appropriate for use in an IR context, where users generally have only a vague picture of the sorts of documents that might satisfy their information need."

On the other hand, Boolean systems are still extremely popular, especially with professional searchers; in fact, there is considerable evidence that, as of mid 1999, professional searchers—librarians and others—preferred Boolean systems by a wide margin. For example, commercial services like Westlaw, DIALOG, and Lexis/Nexis began offering a choice of best-match and exact-match retrieval between late 1992 and early 1994, and the "new way to search old systems [gained] a lot of attention" (Tenopir and Cahn 1994). But it is common knowledge that the vast majority of queries to these services still use the old Boolean interfaces, and therefore exact-match evaluation. In mid-1999, Joanne Claussen of Westlaw said of their service (Claussen 1999): "The usage of Boolean versus natural language varies from group to group and database to database. In general 10 to 20% of searches involve natural language and 80 to 90% [use] Boolean (either directly or through a template). Academic users make significantly greater use of natural language than other users but still remain below 20% of usage." Daniel Pliske of Lexis-Nexis reports similar figures for that service (Pliske 1999). A word of caution: Claussen commented that Boolean is the default for most of Westlaw's databases, so "there is a built-in-bias towards Boolean"; we have not been able to determine whether the same holds for Lexis-Nexis. But it seems unlikely that this built-in bias explains completely the overwhelming popularity of Boolean searching; why is Boolean search still so popular?

It is probable that most professional searchers on the job today have far more experience with, and had far more training with, exact-match than with best-match systems. Hence, an obvious explanation for these searchers' preference for exact-match evaluation is simply that they are more comfortable with what they were trained with and have experience with. While there is undoubtedly some truth to this, it does not explain everything. For one thing, if lack of experience with best-match systems was the only significant factor, one would expect these systems to gradually gain in popularity with expert searchers, but this does not appear to be happening. Claussen (1999) went on to say that "my perception has been that the usage of WIN [Westlaw's 'natural language' interface] has remained fairly consistent since its release."

Of course, situations differ: a searcher might prefer best-match in one case but exact-match in another. Factors that might be important include:

- Do the systems available support exact-match only, best-match only, or both models? (If only one model is supported, of course it's all that people will use.)
- Do the databases available contain free-text fields, or are all text fields controlled-vocabulary (CV)? (The case for Boolean queries is much stronger in CV situations.)
- What types of information does the user need? In particular, how interested are they in precision vis-à-vis recall? (In large databases, Boolean queries are much better at precision than at recall; more to the point, they are arguably better at precision than best-match systems are.)

In mid 1997, we interviewed seven university librarians regarding their views on Boolean vs. best-match searching (Byrd, 1998). Nearly all of them had extensive searching experience. An excerpt from one interview, with the reference librarian at a science branch library, is revealing:

> She graduated from library school 10 years ago; in those days, Boolean was all they taught. All the databases in her library use Boolean searching. She likes Boolean because it's predictable. Alta Vista is maddening because they give you hints ("+ = required") but don't really tell you what they're doing. Also, as an expert searcher, she needs to be able to communicate with users, and Boolean is good for that.

Along the same lines, another of these librarians, one who learned searching with DIALOG over 20 years ago, commented that he had been "experimenting with Alta Vista but often has no idea why it finds documents."

Most of these librarians made comments to the effect that most or all of the databases they had access to were controlled vocabulary.

Finally, a more recent conversation with one of these librarians, one who does reference work, elicited the comment that best-match searching is "smoke and mirrors", and why should she trust a machine more than herself?

In fact, our discussions with librarians confirm our general impression that expert searchers generally say one or both of two things when asked why they use Boolean systems instead of best-match ones: that Boolean evaluation is what they are used to, and/or that they feel much more in control with Boolean systems.

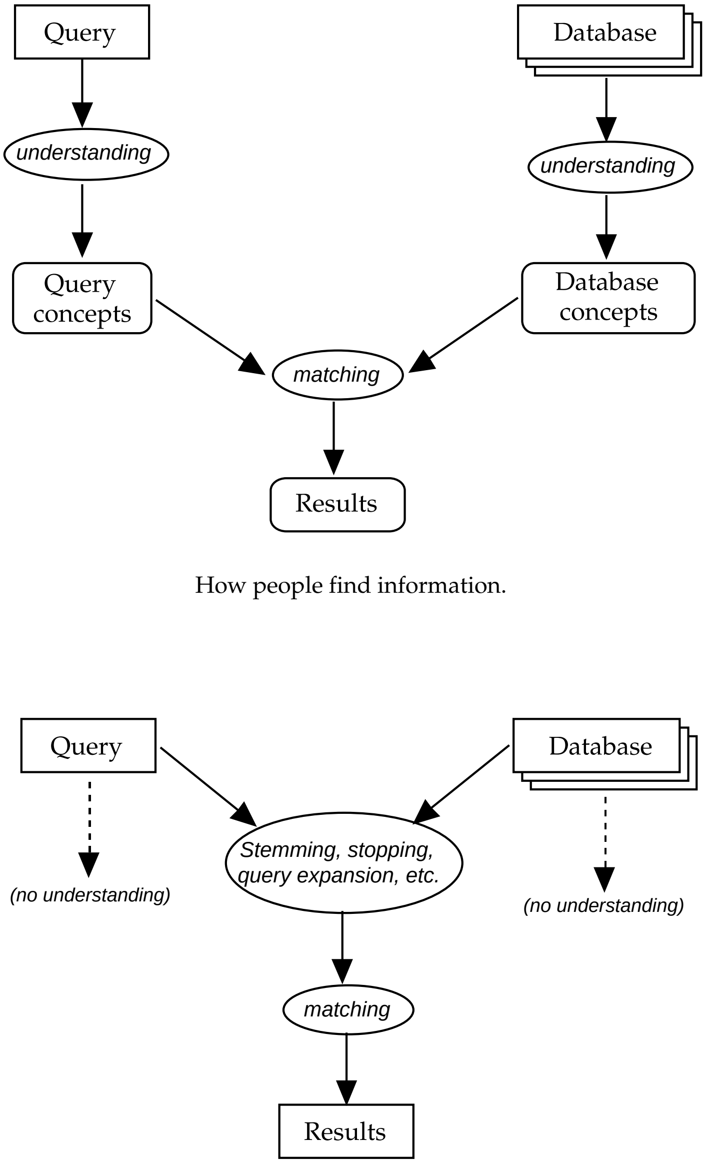
We believe that, with current user interfaces, Boolean retrieval is more *explainable* than best-match in the sense that it is easy to explain why a Boolean system retrieved or didn't retrieve a given document, regardless of its actual relevance. The same cannot be said of a best-match system. For example, with the query "Monarch" AND "butterfly", a Boolean system will retrieve all documents that use both words and no other documents, and it will be obvious why any given document was or was not retrieved. Compare this to giving the roughly-analogous query "Monarch butterfly" to a probabilistic system. Not only is the system likely to retrieve documents that discuss either queens and kings, or butterflies in general, without mentioning Monarch butterflies; but—much worse—it stands a good chance of ranking some of these documents above any document that really does refer to Monarch butterflies. Of course, in this case, proximity restrictions would likely solve the precision problem without hurting recall significantly, but in other cases, they might not.

Is there a way to "have your cake and eat it too", i.e., a way to get the superior control of Boolean retrieval without giving up the greater effectiveness of ranked systems? We believe there is.

2. Hypothesis

Exact-match systems estimate the relevance of each document with simple methods given explicitly by the user. A full understanding of these methods requires only a knowledge of elementary Boolean algebra: to be sure, something beyond most end users. But best-match systems—whether on the probabilistic or vector-space models—are much harder to understand: they estimate relevance with complex algorithms that are barely under user control at all, even if the user is an IR expert. This problem is with what might be called both *active* and *passive* forms of control. Best-match systems lack active control in that the user can't easily influence what they do; they lack passive control in that it is difficult to tell what they are doing. To understand these methods fully requires far more knowledge than do Boolean methods, knowledge perhaps equivalent to a degree in IR. Therefore, user-interface mechanisms that clarify the behavior of these algorithms, and not just their results, are essential.

The distinction between concepts and words underlies all the difficulties of text retrieval. To satisfy the vast majority of information needs, what is important is concepts, but—until they can truly understand natural language—all computers can deal with is words. A diagram can clarify this. Ovals represent processes; rectangular and rounded boxes represent objects. Specifically, rectangular boxes correspond to "well-defined" objects, in this case character strings or documents, while rounded boxes correspond to mental objects, which are never well-defined.



How people find information.

How computers find information.

Figure 1. Finding information: people vs. computers

Concepts are inherently "fuzzy" (we use the term in an informal sense); so is the relationship between a given concept and a given word. But even when someone really does want words (usually for a so-called "known-item" search, i.e., looking for a specific document they already know of), slightly-fuzzy searches may be better. This is because (1) the user might be wrong about one or more of the words, and they or documents in the database might have spelled something differently (either by mistake or as a variant), so a small amount of fuzziness can improve recall considerably; and (2) a little fuzziness is not likely to hurt precision much.

Now, best-match evaluation is inherently fuzzy; exact-match (Boolean) is not. Therefore best match is "better", i.e., it matches human thought better. But the fuzziness makes it harder to understand why certain documents were retrieved but certain others were not, and this makes users uncomfortable. Note that if the results were superb, i.e., if our "magic" were powerful enough, this would not be an issue: users would know that a given document was retrieved because it was relevant, or not retrieved because it was not relevant. But this, of course, is far beyond the state of the art.

What we have said up to this point is independent of query length, but very short queries may be different. It has been pointed out that with two or three terms queries, a document that satisfies the strictest Boolean evaluation—intersecting all the terms—is probably more likely to be relevant than a document that best-match evaluation rates highly but which omits some of the terms.

Therefore, while best-match is the better model, it can be improved for short queries by giving some consideration to Boolean factors. One way to do this is by expanding short queries with combinations of search terms and using Boolean factors to adjust weights on the combinations, as in THOMAS (Croft, Cook, and Wilder, 1995). A more thoroughgoing attempt to combine Boolean and best-match evaluation is described by Rose and Stevens (1997).

Here is a very rough summary, with "letter" grades (A is the best; F means "failure"):

	Small database	*Large database*
Exact-match	Effectiveness: C- Explainability : B	effectiveness: F explainability : B
Best-match	Effectiveness: C Explainability : D	effectiveness: C explainability : D

In other words, for small databases, exact match searching is nearly as effective as best match, and it is much more explainable. For large databases, exact match is so ineffective, its advantage in explainability is worthless; this is unfortunate, because the explainability of best match is barely acceptable.

We hypothesize that *appropriate visualizations can improve the explainability of best-match systems enough that both expert and ordinary users will prefer them over exact-match systems.*

Note that different standards must be used to judge the success of an interface for experts and for "ordinary" users. For experts, we want to establish that a best-match system that gives users more control than existing best-match systems can be as satisfying as existing Boolean systems. For average users, a more appropriate standard of comparison is the type of interface the major Web-search services provide in their usual (i.e., not "advanced") modes.

3. Approaches to Visualizing IR

To verify the above hypothesis, we identified a set of information aspects whose visualization seems likely to improve the best-match system's explainability. Next we designed a software system, MIRV (Multiple Information Retrieval Visualizations), that employs visualizations throughout. We implemented parts of MIRV and ran an experiment to determine whether the explainability of the system was indeed improved. This section will briefly overview the set of information aspects and the features of the visualizations.

In principle, an interface that gives users substantially more control over best-match searches should be useful to the average person as well as to the expert searcher. However, different groups of users have different needs (Shneiderman 1998, pp. 67–70). As a result, optimal implementation choices vary significantly from user to user. Furthermore, as we have said, different standards must be used to judge the success of interfaces for experts and for "ordinary" users.

It might be objected that no visualization can by itself give users more control, since—even if the visualization provides just the information a user wants—there might be no effective way for them to take advantage of that knowledge. This is a strong objection, and in fact we do provide some means of active control (see definition above). But we also believe that more information about what is happening *in itself* gives users more control, assuming that that information is presented in a way they can understand. If nothing else, greater understanding of what an IR system is doing with a user's query might help a user decide more quickly that they will not be able to find the information they want and that, therefore, they should abandon the attempt. And the desirability of giving users a feeling of control is a truism among workers in human/computer interaction. In the words of Hix and Hartson 1993 (pp. 32–33), "Keep the locus of control with the user".

Several aspects of the information involved might be visualized. A minimal list of sensible visualizations, with their primary application phases *in italics,* might look like this (phases are named in the terms of Shneiderman et al 1997 and Shneiderman et al 1998):

Vquery. the query alone: *formulation*

Vquery-DB. the query in relation to the database(s): *formulation*

Vquery-doc. the query in relation to individual retrieved documents: *review of results*

Vdoc-doc. the retrieved documents in relation to each other: *review of results*

Each of these visualizations can be done in many ways. First, in most cases, different pieces of information might be visualized. For instance, Vquery-doc might show the numbers of occurrences of each query term (as in the commercial system CALVIN), the contributions of each term to the document's score (as in Shneiderman et al 1997, Fig. 2), or the progression of appearances of terms in the document (as in Hearst's tilebars: see Hearst 1995). At its most basic, it might give term-occurrence information in Boolean form simply by listing the terms that appear in the document at all (as in PRISE: see Harman 1992).

Second, there are various ways to realize a visualization of given information graphically, some simpler, some more complex. For example, Vdoc-doc might be realized in either 2-D or 3-D.

Third, while the term "visualization" suggests a passive display, visualizations can also be interactive, with affordances to let the user control the system. In our case, it is certainly possible (and it may well be desirable) to offer control of the full query-expressing power of a modern IR system in the framework of Vquery and Vquery-DB.

Finally, for performance reasons, one might prefer to visualize an approximation to the desired information. For Vquery-DB in particular, one can show the query against the actual databases to be used, or against a "proxy" *query-formulation database.* This is a special local "database"—actually, a database index alone, without associated documents—whose terms and term frequencies approximate those of the databases to be searched. Naturally, it is preferable to use the actual databases, but the proxy is often much more practical, especially in a client/server situation (and most especially on the World Wide Web). This is essentially the "query previews" idea of Doan et al (1996). The query-formulation "database" contains no document text, and its index does not need location information; for both reasons, it can be small enough to fit easily on a local disk.

Also note that some of these visualizations might be more or less tightly integrated: for example Vquery and Vquery-DB could be shown on a single set of axes.

MIRV employs all of these visualizations except Vdoc-doc; in fact, it employs two versions each of some. We now outline MIRV in some detail.

A Note on the MIRV Project

As we have said, MIRV was originally envisaged as adding appropriate visualizations at every stage of the text-retrieval task. Unfortunately, only a subset of it has been implemented, and in that subset, the implementation differs from the original design. Also, only a subset of the subset has undergone formal usability testing. As a result, this paper first describes and illustrates the original design; then it describes the implementation. Finally it describes testing thus far, and plans for testing the entire MIRV system when it is implemented.

4. MIRV: The Original Design

Assumptions

We assume:

- Everything is in English: databases, queries, and user interface.
- Users are adult native speakers of English, with normal color perception and at least some experience with computers. In terms of experience with online searching, they may run the gamut from novices to experienced amateurs to experts.
- Environment:
    - Either standalone, or client/server with a high-bandwidth link, e.g., Ethernet. Modem speed (56 Kbps at the very best) may not be adequate.
    - Significant client-side software, e.g., Java programs.
    - Good client-side hardware, e.g., Pentium 200 or better, and 800x600-pixel or better color monitor.
- We will support what might be called Boolean and probabilistic types of desirable terms—"required" and "preferred" terms, respectively—but only the probabilistic type of undesirable terms—"avoid" terms. This is because Boolean undesirable

terms ("forbidden" ones) "are rarely useful in practice: much more often, they simply give users a way to hurt themselves by lowering recall without necessarily gaining precision" (Allan 2000).

- NLP-based techniques will be used, but only to a very limited extent: most of these techniques are not robust enough for our purposes. Considering such clues as punctuation, capitalization, and use of prepositions, we expect NLP to perform with reasonable accuracy part-of-speech tagging; recognize names of organizations and individuals; and identify words that act as Boolean operators ("and", "or", "not", "but", etc.) and their scopes. We might be able to use NLP on a query like "Lafayette's role in the American Revolution" to recognize the vital importance of "Lafayette" in relevant documents. We might also be able to use NLP to find quotations and identify who they are by and/or about, but it is doubtful that the effort would be worthwhile for a general-purpose IR system.
- In addition to indexing, for performance reasons, databases may undergo preprocessing for NLP and perhaps for identification of information needs they are suitable for, i.e., "collection selection".

Note the implications for the World Wide Web of the requirements for high-bandwidth links and for control of databases: the bandwidth requirement excludes the Web in many situations, but the database-control requirement excludes it entirely.

User Interaction as Originally Designed

MIRV uses a conventional GUI interface. Most interaction takes place in the *query entry window* or the *query interpretation window*.

Figs. 1 through 6 are snapshots of possible interaction as a user formulates a query for a single information need, runs it, and looks at the documents retrieved. The user here is interested in information about multimedia information retrieval, but they are less interested if the information pertains to IBM.

The concept of a query-formulation database, henceforth called a "QF" database, has already been mentioned, and the discussion below assumes one.

The Figures

Recent versions of standard word-processing programs like Microsoft Word finally include well-thought-out devices to give users control of potential typos, especially by drawing attention to suspicious "words" as soon as they're typed with a wavy red underline. Our situation is similar and we adopt the same UI.

Fig. 1 shows the system's initial display. In **Fig. 2**, the user has entered the query "multimedia retrieval but not any IBM reserach" in the query-entry window. The system indicated with the wavy red underline that "reserach" does not appear in the QF database: this suggests a possible typo or other problem. However, the user ignored this feedback and chose the Interpret button; the system responded with the two visualizations in the query interpretation window shown on the bottom: Vquery (labelled "QUERY") above, Vquery-DB-hat (labelled "QUERY AND DATABASES") below. Vquery-DB-hat is a query-formulation database. Since Vquery-DB-hat is shown rather than Vquery-DB and the difference can be important, the word "Estimated" appears prominently. Note that the bars in Vquery are of unequal lengths, reflecting the system's estimates of the corresponding terms' importance (in a typical best-match system, log *idf*), and are labelled accordingly. Both visualizations are interactive. Vquery provides control simply by making it easier for users to see and (by dragging the endpoints of bar segments) adjust the relative importance assigned to components of the query.

Vquery-DB-hat shows approximately how well the query terms match the databases: the number next to each indexed term is its total frequency in the database. Simply seeing clearly how many occurrences the databases have of each phrase, term, synonym, and variant can greatly help users in avoiding problems of vocabulary mismatch, of spelling errors, and of query terms that—because of their high frequency—are of little value in discriminating among documents. However, this visualization also affords control over several aspects of the query, specifically:

1. Variants conflated by stemming. To see and change treatment of variants, click on one of the horizontal triangles: see discussion of Fig. 3 for details.
2. Words that do not appear in the QF database and will therefore—depending on how accurately the QF database reflects the actual databases—probably be ignored when the search is run. These words fall into two categories: those that appear in a list of stop words ("STOPPED"), and those that are neither stopped nor indexed (frequency "ZERO").
3. Relative weights of occurrences of different terms. In Vquery, the horizontal extent of each bar segment shows the effective term weight that will be used in evaluating documents: the contributions of the weight assigned by the user and the weight assigned by the system (typically log *idf*) are not differentiated. (This might be called the "standard view" of Vquery. We have also considered an "expert view", in which the horizontal extent of each bar segment shows the user weight, and the vertical extent shows the system-assigned weight. The product of these—graphically, the segment's area—is the effective weight. However, it is not clear that this distinction would often be useful for even expert searchers, though it probably would be for IR researchers.)

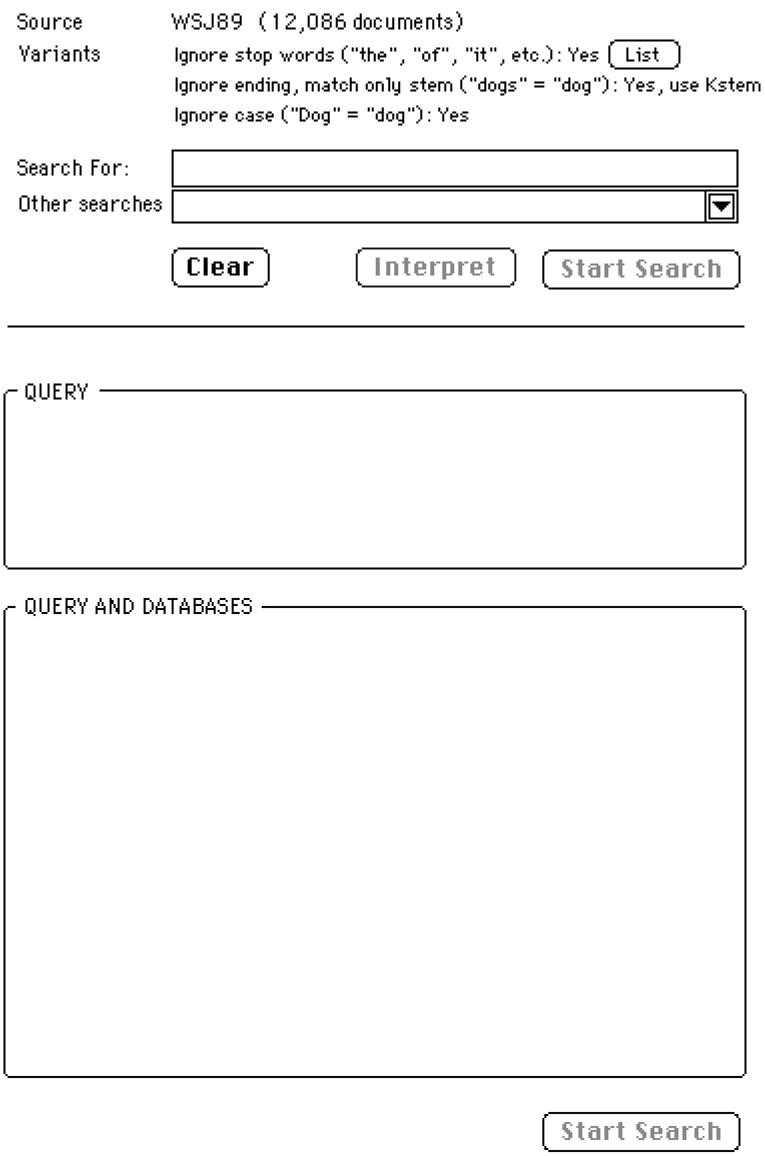
In **Fig. 3**, the user has clicked on a horizontal triangle to see the variants of "retrieval". The system replaced the horizontal triangle with a vertical triangle and added to the display a list of variants in the QF database, with their frequencies. The user can select items from this list and remove them: in the figure, "retrieving" is selected. Note that if stemming is done at indexing time, information on what variants of a given term appear in the databases will be extremely difficult to get, so control this fine-grained is probably not practical. Also note that a given raw term may map to different stems in different databases; we have not addressed this concern.

The user next adds the term "interface", corrects the spelling of "research" and makes it DESIRABLE instead of UNDESIRABLE, and tells the system to consider "multimedia retrieval" a phrase: the new query might be expressed as "'multimedia retrieval' interface research but not IBM". Then they add synonyms for several words, with the result shown in **Fig. 4**. Now the "QUERY AND DATABASES" box depicts a more complex situation, including synonyms and phrases as well as variants conflated by stemming. Words listed vertically next to a single bar are synonyms, for example "IBM" and "Big Blue", or "retrieval", "search", and "index".

In **Fig. 5**, the user has actually run the query (by clicking Start Search). The system responded by showing the beginning of a result list, with a form of Vquery-doc ("Vquery-doc1") to its left. (This visualization is briefly discussed in Shneiderman at al 1997.) For each document in the result list, the best passage is shown in a slightly "fisheye" view, i.e., text lines after the first are shown in smaller and smaller sizes. Clicking on one of the colored bar segments in Vquery-doc1 shows the best passage in the document for that term. This display supports user control by:

1. Showing the relative contributions of query terms to document scores. In the "Monarch butterfly" case, the retrieval engine might rate some documents with only one or two mentions of "Monarch" but 50 mentions of "butterfly" above documents with comparable numbers of each. The user would probably be much more interested in the more balanced documents, and it would be obvious at a glance which were which.
2. Automatically showing the best overall passage for each document. IR systems generally show the beginning of each document in a result list, but (both intuitively and according to such recent studies as Tombros and Sanderson 1998) the best passage is a much better answer to the questions "Why was this document ranked as it was?" and "Why was this document retrieved at all?"
3. Letting the user easily view the best passage for each term. This should help determine whether the document discusses the desired concepts, regardless of whether the requested words are used.

In Fig. 6, the user has opened a document from the result list; it is displayed in a separate window in the form shown. The vertical scrollbar contains a second form of Vquery-doc. The colored icons (with matching color highlighting) for occurrences of query terms in this "Vquery-doc2" give the same information as Hearst's Tilebars, but at no cost in screen "real estate", and in a form that should let users browse through documents as efficiently as possible. In fact, Vquery-doc2 should help the user determine whether the document discusses the desired concepts with far more confidence than either Vquery-doc1 or tilebars do. If the document does in fact discuss those concepts, Vquery-doc2 should also help determine whether it discusses the concepts *in relation to each other* with at least as much confidence as tilebars, and with much more confidence than Vquery-doc1.



Source WSJ89 (12,086 documents)

Variants Ignore stop words ("the", "of", "it", etc.): Yes

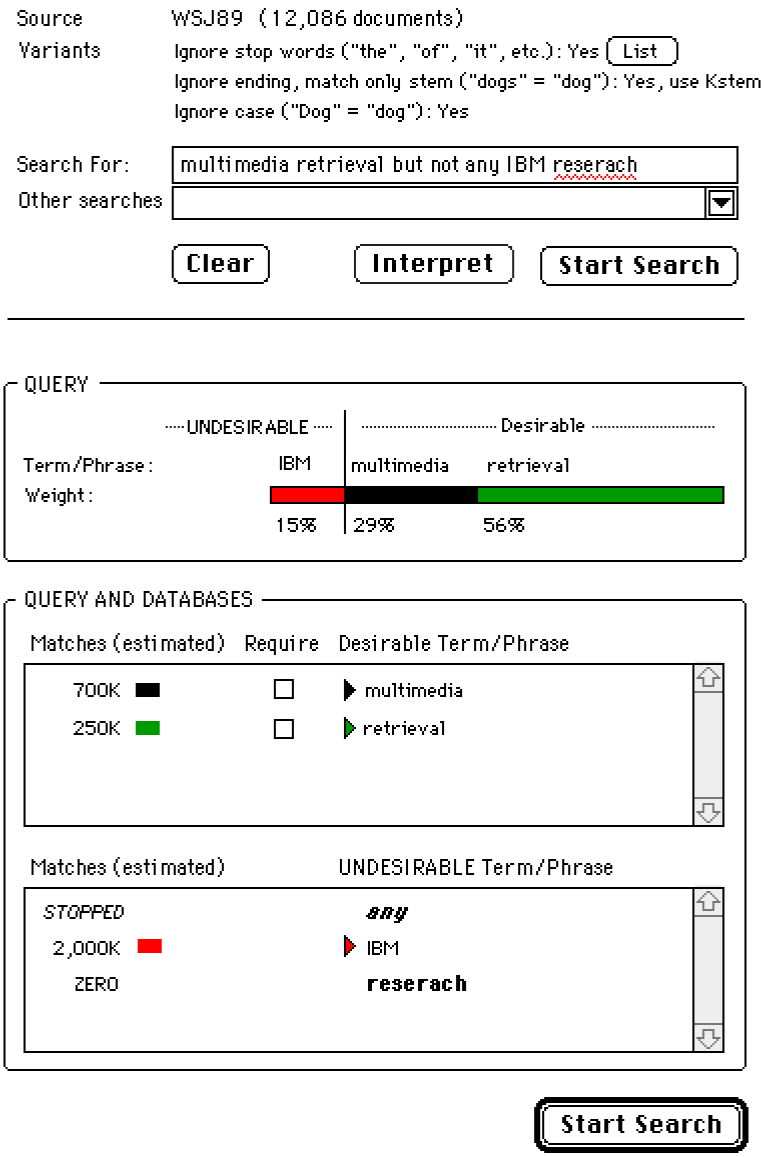
Ignore ending, match only stem ("dogs" = "dog"): Yes, use Kstem

Ignore case ("Dog" = "dog"): Yes

Search For:

Other searches

Figure 1 (original design). Initial Display (on startup)



Source WSJ89 (12,086 documents)

Variants Ignore stop words ("the", "of", "it", etc.): Yes List
Ignore ending, match only stem ("dogs" = "dog"): Yes, use Kstem
Ignore case ("Dog" = "dog"): Yes

Search For: multimedia retrieval but not any IBM reserach

Other searches

QUERY

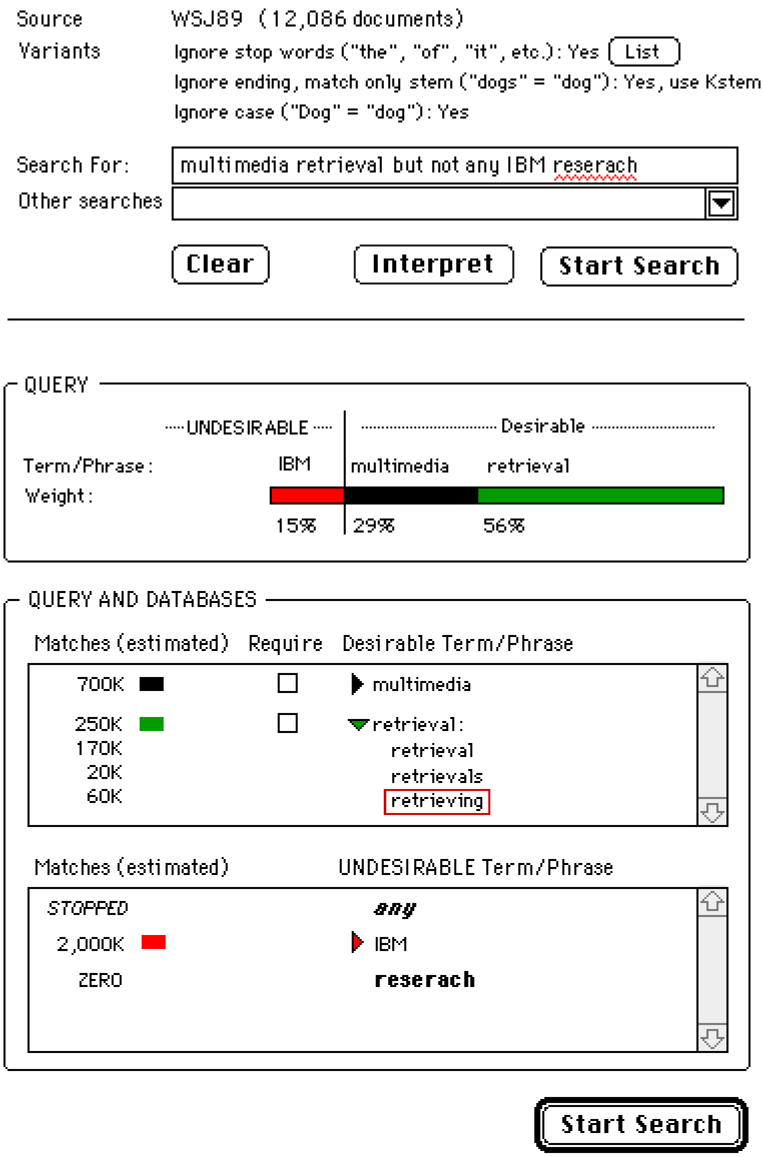
	UNDESIRABLE	Desirable	
Term/Phrase:	IBM	multimedia	retrieval
Weight:	🟥	■	🟩
	15%	29%	56%

QUERY AND DATABASES

Matches (estimated)	Require	Desirable Term/Phrase
700K ■	□	▶ multimedia
250K 🟩	□	▶ retrieval

Matches (estimated)	UNDESIRABLE Term/Phrase
STOPPED	***any***
2,000K 🟥	▶ IBM
ZERO	**reserach**

Figure 2 (original design). Start: user types/pastes/chooses a query, clicks Interpret => Vquery and Vquery-DB-hat



Source	WSJ89 (12,086 documents)
Variants	Ignore stop words ("the", "of", "it", etc.): Yes
	Ignore ending, match only stem ("dogs" = "dog"): Yes, use Kstem
	Ignore case ("Dog" = "dog"): Yes
Search For:	multimedia retrieval but not any IBM reserach
Other searches	

QUERY

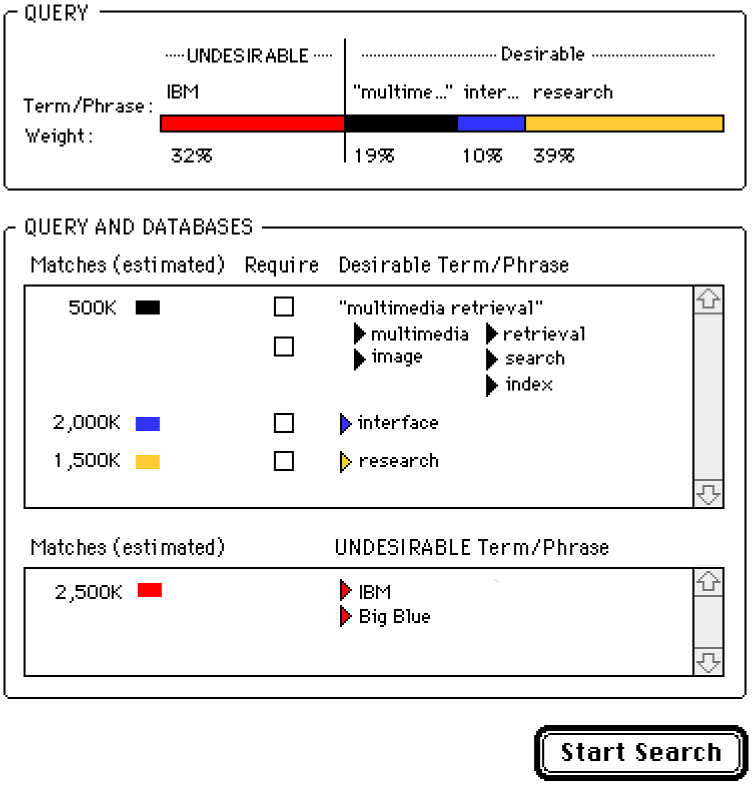
	UNDESIRABLE	Desirable	
Term/Phrase:	IBM	multimedia	retrieval
Weight:	15%	29%	56%

QUERY AND DATABASES

Matches (estimated)	Require	Desirable Term/Phrase
700K ■	□	▶ multimedia
250K 🟩	□	▼ retrieval:
170K		retrieval
20K		retrievals
60K		retrieving

Matches (estimated)	UNDESIRABLE Term/Phrase
STOPPED	***any***
2,000K 🟥	▶ IBM
ZERO	**reserach**

Figure 3 (original design). User clicks the triangle before "retrieval" to see variants (presumably from stemming)

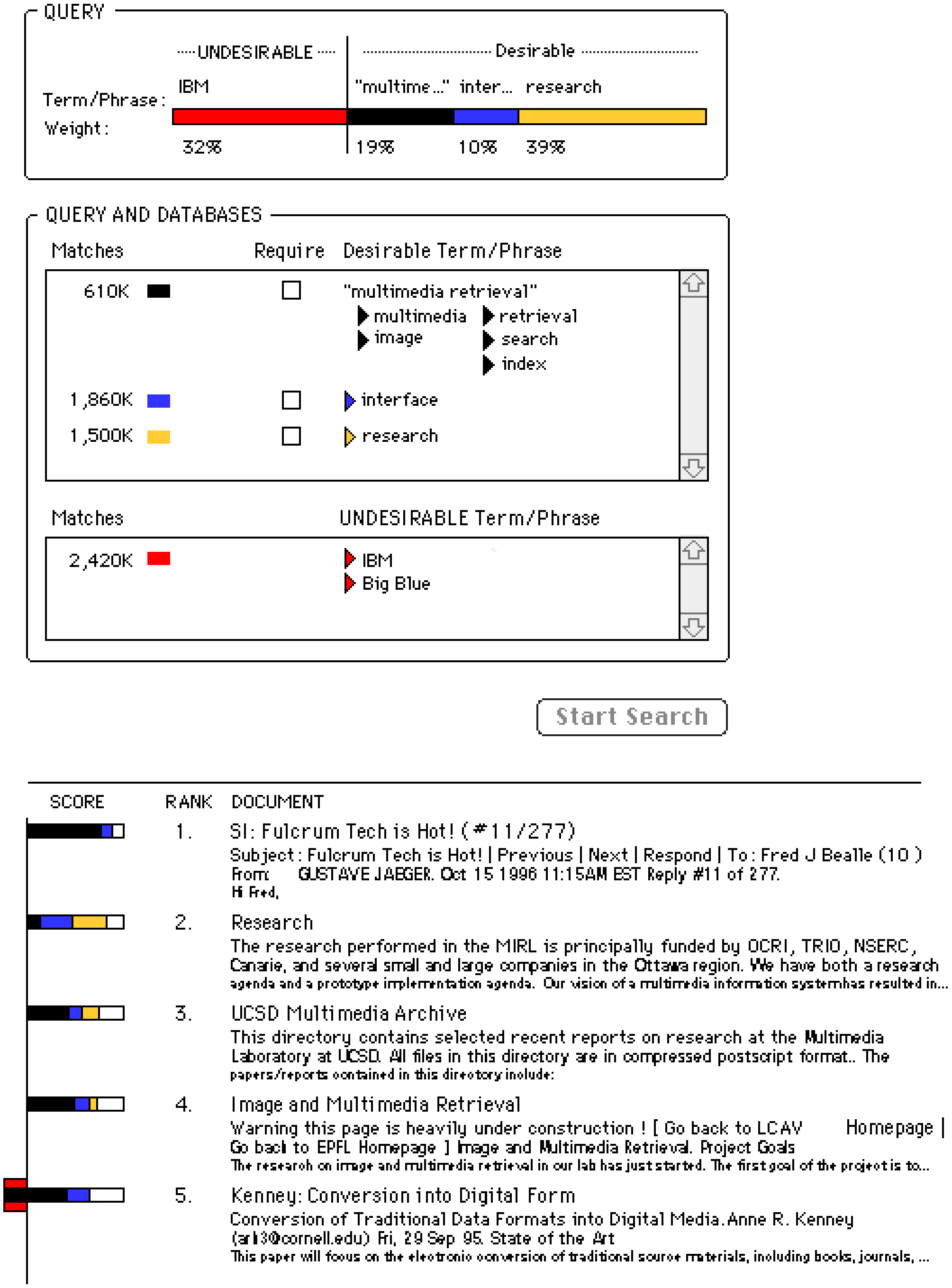


	UNDESIRABLE	Desirable		
Term/Phrase:	IBM	"multime..."	inter...	research
Weight:	32%	19%	10%	39%

Matches (estimated)	Require	Desirable Term/Phrase
500K ■	□	"multimedia retrieval"
	□	▶ multimedia ▶ retrieval ▶ image ▶ search ▶ index
2,000K 🟦	□	▶ interface
1,500K 🟨	□	▷ research

Matches (estimated)	UNDESIRABLE Term/Phrase
2,500K 🟥	▶ IBM ▶ Big Blue

Figure 4 (original design). User makes "multimedia retrieval" a phrase, adds words to the query, corrects "reserach", and adjusts weights



1. SI: Fulcrum Tech is Hot! (#11/277)

Subject: Fulcrum Tech is Hot! | Previous | Next | Respond | To: Fred J Bealle (10)
From: GUSTAVE JAEGER. Oct 15 1996 11:15AM EST Reply #11 of 277.
Hi Fred,

2. Research

The research performed in the MIRL is principally funded by OCRI, TRIO, NSERC, Canarie, and several small and large companies in the Ottawa region. We have both a research agenda and a prototype implementation agenda. Our vision of a multimedia information systemhas resulted in...

3. UCSD Multimedia Archive

This directory contains selected recent reports on research at the Multimedia Laboratory at UCSD. All files in this directory are in compressed postscript format.. The papers/reports contained in this directory include:

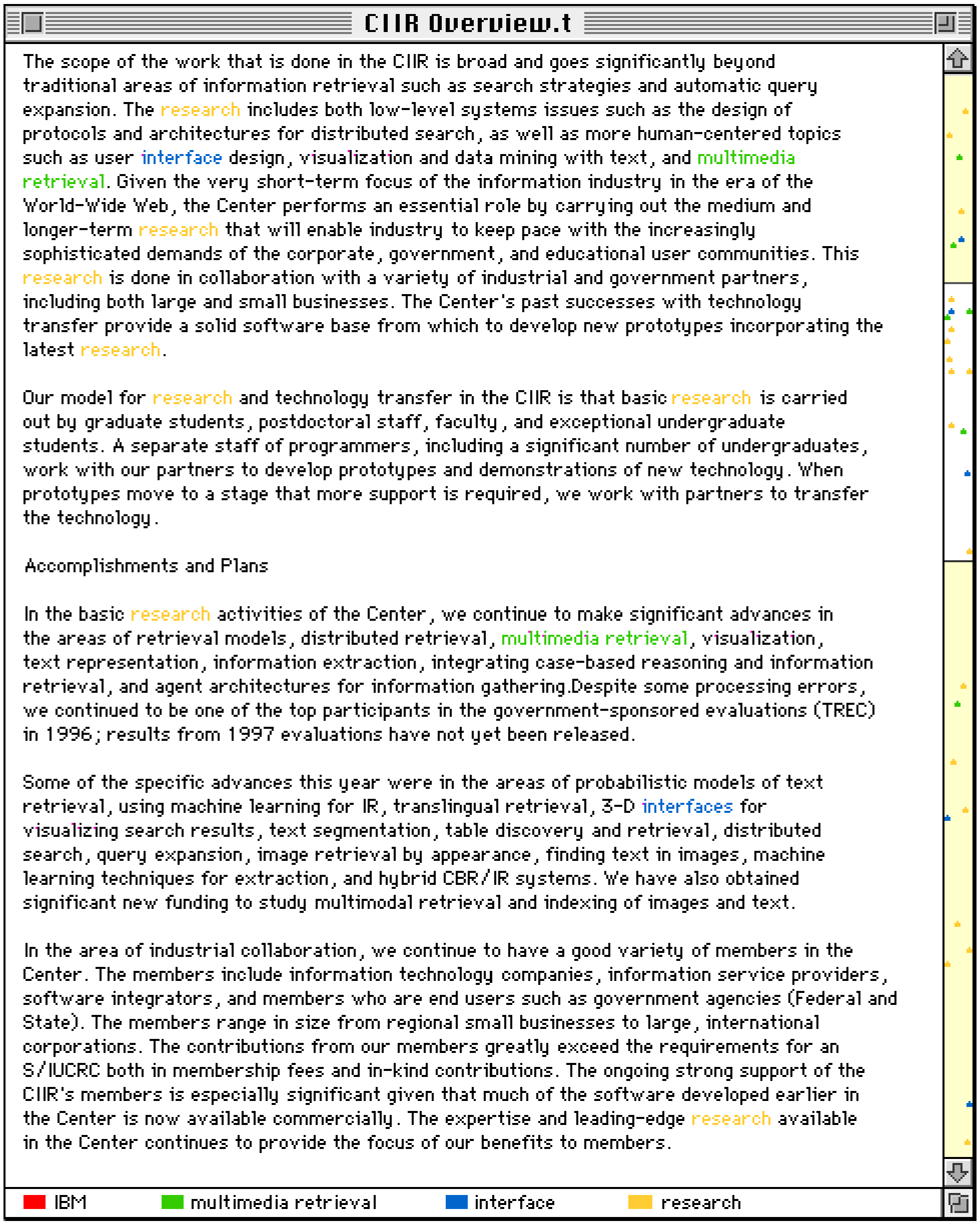
4. Image and Multimedia Retrieval

Warning this page is heavily under construction ! [Go back to LCAV Homepage | Go back to EPFL Homepage] Image and Multimedia Retrieval. Project Goals
The research on image and multimedia retrieval in our lab has just started. The first goal of the project is to...

5. Kenney: Conversion into Digital Form

Conversion of Traditional Data Formats into Digital Media. Anne R. Kenney (arl3@cornell.edu) Fri, 29 Sep 95. State of the Art
This paper will focus on the electronic conversion of traditional source materials, including books, journals, ...

Figure 5 (original design). User runs the query => Vquery-DB-hat is replaced with Vquery-DB, and a result list appears, showing Vquery-doc1



The scope of the work that is done in the CIIR is broad and goes significantly beyond traditional areas of information retrieval such as search strategies and automatic query expansion. The research includes both low-level systems issues such as the design of protocols and architectures for distributed search, as well as more human-centered topics such as user interface design, visualization and data mining with text, and multimedia retrieval. Given the very short-term focus of the information industry in the era of the World-Wide Web, the Center performs an essential role by carrying out the medium and longer-term research that will enable industry to keep pace with the increasingly sophisticated demands of the corporate, government, and educational user communities. This research is done in collaboration with a variety of industrial and government partners, including both large and small businesses. The Center's past successes with technology transfer provide a solid software base from which to develop new prototypes incorporating the latest research.

Our model for research and technology transfer in the CIIR is that basic research is carried out by graduate students, postdoctoral staff, faculty, and exceptional undergraduate students. A separate staff of programmers, including a significant number of undergraduates, work with our partners to develop prototypes and demonstrations of new technology. When prototypes move to a stage that more support is required, we work with partners to transfer the technology.

Accomplishments and Plans

In the basic research activities of the Center, we continue to make significant advances in the areas of retrieval models, distributed retrieval, multimedia retrieval, visualization, text representation, information extraction, integrating case-based reasoning and information retrieval, and agent architectures for information gathering.Despite some processing errors, we continued to be one of the top participants in the government-sponsored evaluations (TREC) in 1996; results from 1997 evaluations have not yet been released.

Some of the specific advances this year were in the areas of probabilistic models of text retrieval, using machine learning for IR, translingual retrieval, 3-D interfaces for visualizing search results, text segmentation, table discovery and retrieval, distributed search, query expansion, image retrieval by appearance, finding text in images, machine learning techniques for extraction, and hybrid CBR/IR systems. We have also obtained significant new funding to study multimodal retrieval and indexing of images and text.

In the area of industrial collaboration, we continue to have a good variety of members in the Center. The members include information technology companies, information service providers, software integrators, and members who are end users such as government agencies (Federal and State). The members range in size from regional small businesses to large, international corporations. The contributions from our members greatly exceed the requirements for an S/IUCRC both in membership fees and in-kind contributions. The ongoing strong support of the CIIR's members is especially significant given that much of the software developed earlier in the Center is now available commercially. The expertise and leading-edge research available in the Center continues to provide the focus of our benefits to members.

Figure 6 (original design). Document viewer, showing Vquery-doc2

The MIRV UI Paradigm

The sequence of user interaction underlying the above scenario is as follows:

a. The user enters a query in free-text form (Figs. 1 and 2).

b. The system optionally *interprets* the query and responds with a direct-manipulation visualization of the query in the abstract (Vquery), and a passive visualization of the query against the actual database(s) to be used (Vquery-DB) or against a query-formulation database (Vquery-DB-hat) (Fig. 2).

c. The user refines the query by editing the free text, in which case the system updates the visualization, or by editing the visualization directly. (Figs. 3 and 4.) In the latter case, the system probably does *not* respond by updating the free text: in general, that would be extremely difficult to do. This is especially true because, if the system were to update it, it would be very desirable to minimize user disorientation by changing as little as possible of the user's original version, but that makes the task even more difficult.

d. The user *starts the search,* and the system displays a summary of results with a "condensed" visualization of the query against each retrieved document (Vquery-doc) giving the same information as in Fig. 2 of Shneiderman et al 1997 (see our Fig. 5). It may also show a simple visualization of the retrieved documents in relation to each other (Vdoc-doc), e.g., via self-organizing map. If it can be done quickly and was not done in step 2, the system also shows a visualization of the query against the actual database(s) used (Vquery-DB).

e. The user browses retrieved documents, each of which is displayed with an accompanying detailed visualization against the query giving the same information Tilebars give (Vquery-doc), and refines the query with techniques like relevance feedback (Fig. 6).

Note the possibility of using *dynamic-query* techniques (Shneiderman et al 1997). The system might well be able to do step b as the user types, and it might also be able to do step d as the user types and/or as they edit. But doing so is desirable only if system response is fast enough: this is no small concern. The above figures depict a conventional "static query" UI, in which no action is taken until the user specifically asks for it. After entering a query, evidently the user can choose either Interpret to see a visualization, or they can immediately Start Search. But the UI could support dynamic queries, both for Interpret and for Start Search. Most likely the user interface for choosing the static or the dynamic interface would simply be a pair of choices in a Preferences command: if dynamic queries is selected for either choice, the Interpret or Start Search button would be either hidden or disabled. Given a local query-formulation database, running the Interpret process dynamically should work well, but—for response-time reasons—running Start Search dynamically will probably not be practical in many environments.

More important, these designs probably will not work well with large queries containing more than, say, six terms; such queries are unlikely to be entered directly by a user but they could easily result from query-expansion techniques like relevance feedback, LCA, etc. We leave this issue for later study.

5. MIRV Implementation

The Center for Intelligent Information Retrieval's InQuery retrieval engine is written in C; more recently, the Center has developed JITRS (for Java InQuery Text Retrieval System), a Java class library that uses the JNI (Java Native Interface) package to allow Java programs to communicate with InQuery on a client/server basis. JITRS also includes a "default" text-retrieval client, "JDC", that employs the Swing package (part of Sun's Java Foundation Classes) for its user interface. JITRS was designed according to the well-known Model/View/Controller (MVC) paradigm. The MVC design makes it easy to modify any component without affecting the others.

We implemented a subset of MIRV on top of JITRS and of JDC, writing in Java and using Swing extensively. Swing contains an object-oriented GUI toolkit, and the capability it offered of overriding methods of GUI objects greatly eased implementation of one visualization (see Byrd 1999). For simplicity, the initial version of the system does not support any form of query expansion (relevance feedback, LCA, etc.); however, it has been designed so that query expansion can easily be added.

In implementing the MIRV subset, we made several changes to the visualizations: these are described below.

The GUI also uses components from JDC that let the user choose an active database and that show the returned documents (both in a result list and in a separate text-display area). These components correspond to visualization Vquery-doc (the query in relation to individual retrieved documents).

User Interaction as Implemented: An Example

This section briefly describes a scenario of the system's use. See "User Interaction as Originally Designed" and "The MIRV UI Paradigm" above for background. The visualization of the query alone (Vquery) has been broken into two separate components: Vquery1 (Fig. 7) and Vquery2 (Fig. 8). The first component includes the text-input area and a pop-up list of previously entered queries for quick access (Vquery1). It also contains three buttons that allow clearing the input area, starting search, or visually depicting search parameters for the current query.

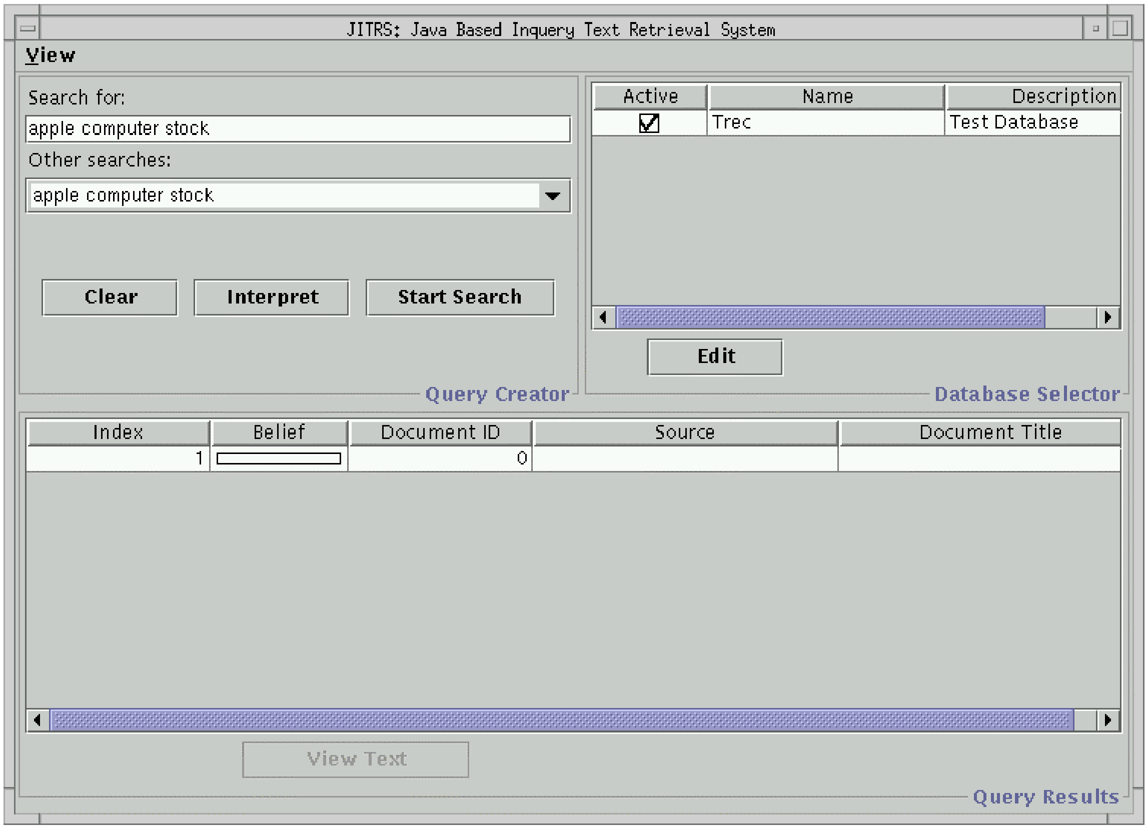
The second component (Vquery2, Fig. 8) graphically depicts weights assigned to terms of the current query. The weights of the other terms of the query are changed correspondingly so that their combined weight stays at 100.

First, the user enters a query, in this case "apple computer stock", into a text area in the Vquery1 component (Fig. 7). Next the user chooses the Interpret button; the system responds with the two visualizations: Vquery2 above (Fig. 8), Vquery-DB below (Fig. 9). Both visualizations are interactive. Vquery2 provides control simply by making it easier for users to see and (by dragging the endpoints of bar segments) adjust the relative importance assigned to components of the query.

In Fig. 9, the user has clicked on a horizontal triangle to see the variants of "stock". The system replaced the horizontal triangle with a vertical triangle and added to the display a list of variants in the QF database: in the implementation, frequencies of variants are not shown. The user can select items from this list and remove them by pressing the Delete key: in the figure, “stockings” is selected. Note that if stemming is done at indexing time, information on what variants of a given term appear in the databases will be very difficult to get, and in that case control this fine-grained is probably not practical. Also note that a given raw term may map to different stems in different databases; we have not addressed this concern.

In Fig. 10, the user has actually run the query (by clicking Start Search). The system responds by showing the beginning of a result list. As we have said, MIRV’s result list is intended to have a form of query/document visualization (“Vquery-doc1”) to the left of the result list, but we have not yet not implemented it.

Finally, in Fig. 11, the user has opened a document from the result list; it is displayed in a separate window in the form shown. The vertical scrollbar contains the second form of query/document visualization, “Vquery-doc2”, with colored query terms and confetti scrollbar.



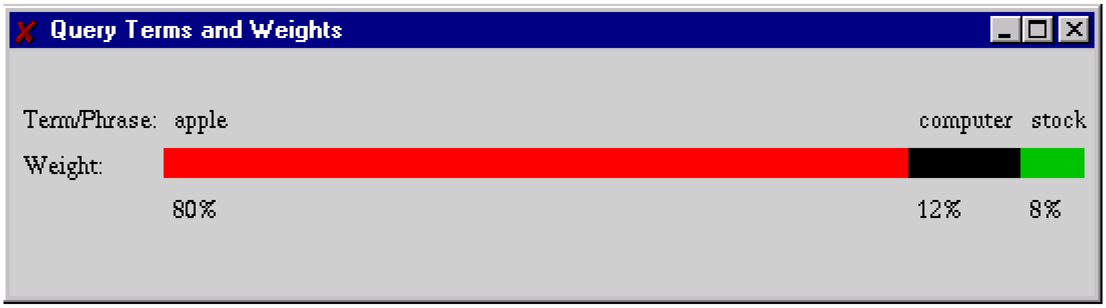
Search for: apple computer stock

Other searches: apple computer stock

Active	Name	Description
☑	Trec	Test Database

Index	Belief	Document ID	Source	Document Title
1	▭	0		

Figure 7 (implementation). Combination of query input area (with Vquery1) and empty result list

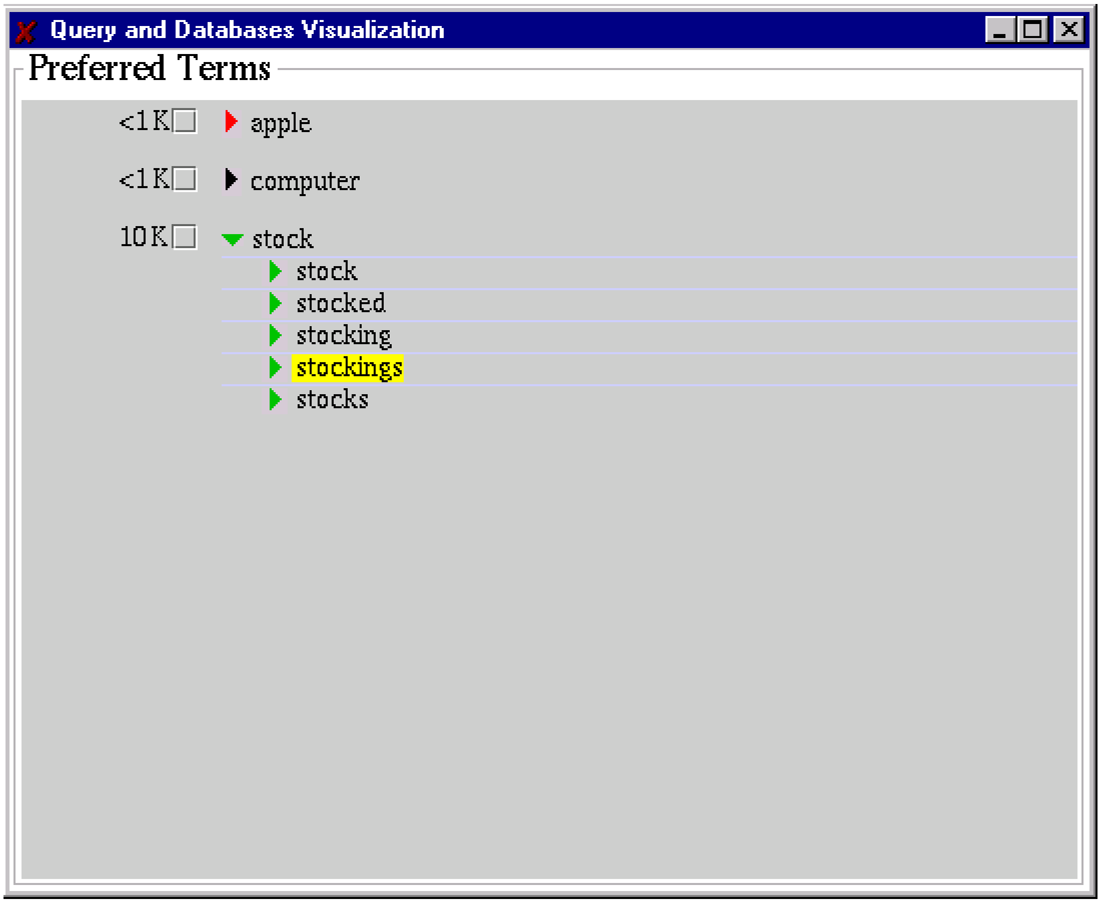


Query Terms and Weights

Term/Phrase: apple computer stock

Weight: 80% 12% 8%

Figure 8 (implementation). Visualization of term weights (Vquery2)



Query and Databases Visualization

Preferred Terms

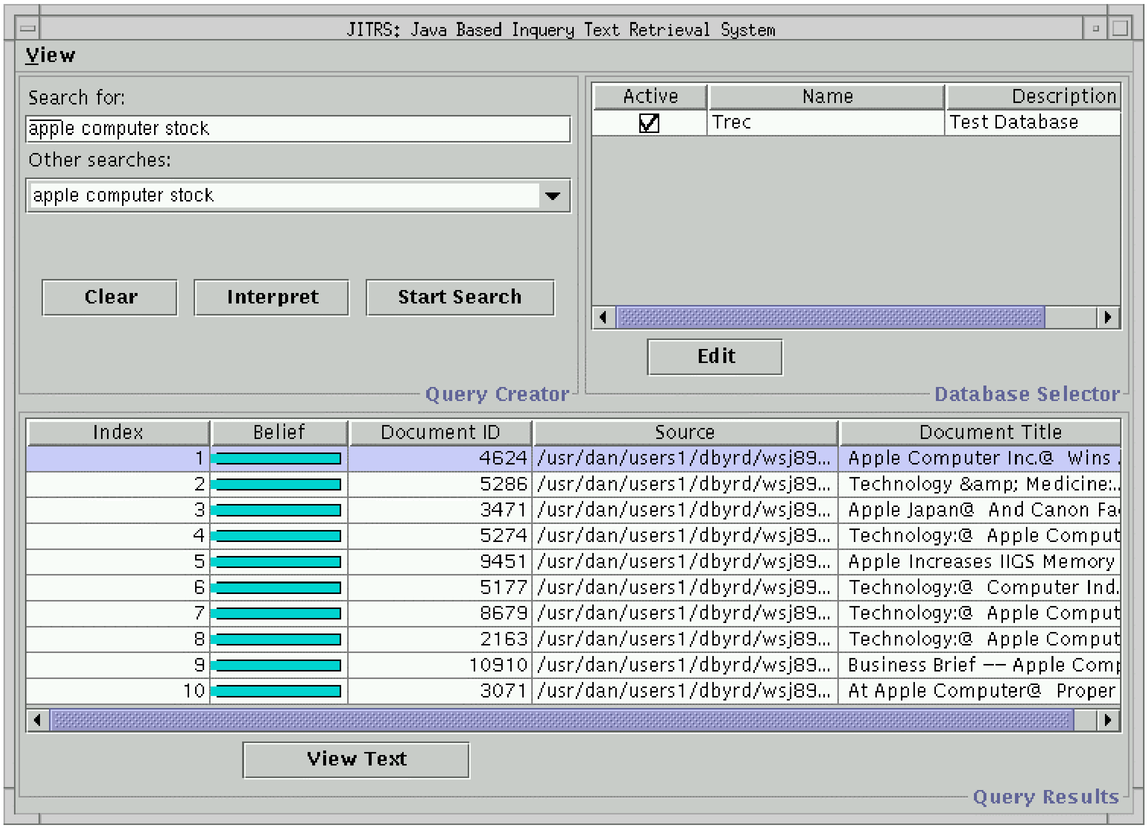
<1K □ apple

<1K □ computer

10K □ stock

- stock
- stocked
- stocking
- stockings
- stocks

Figure 9 (implementation). Visualization of term frequences and variants (Vquery-DB)



Search for:

apple computer stock

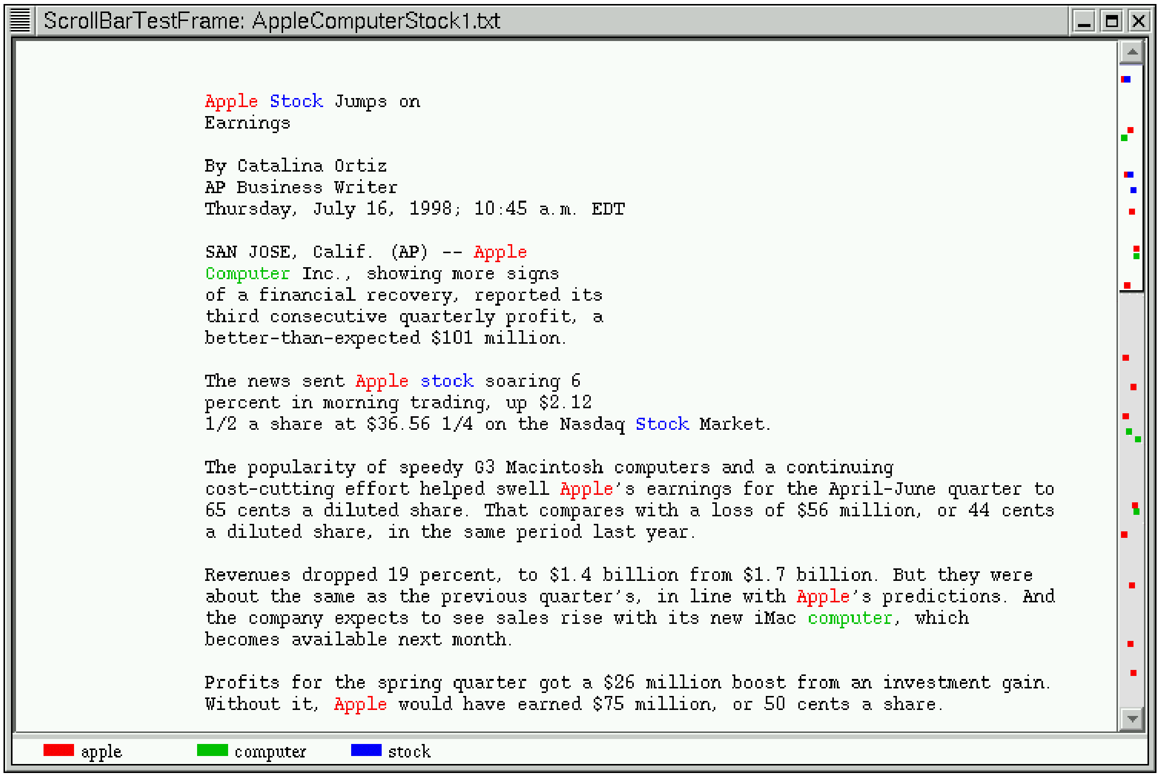
Other searches:

apple computer stock ▾

Active	Name	Description
☑	Trec	Test Database

Index	Belief	Document ID	Source	Document Title
1	▬	4624	/usr/dan/users1/dbyrd/wsj89...	Apple Computer Inc.@ Wins
2	▬	5286	/usr/dan/users1/dbyrd/wsj89...	Technology &amp; Medicine:..
3	▬	3471	/usr/dan/users1/dbyrd/wsj89...	Apple Japan@ And Canon Fa
4	▬	5274	/usr/dan/users1/dbyrd/wsj89...	Technology:@ Apple Comput
5	▬	9451	/usr/dan/users1/dbyrd/wsj89...	Apple Increases IIGS Memory
6	▬	5177	/usr/dan/users1/dbyrd/wsj89...	Technology:@ Computer Ind.
7	▬	8679	/usr/dan/users1/dbyrd/wsj89...	Technology:@ Apple Comput
8	▬	2163	/usr/dan/users1/dbyrd/wsj89...	Technology:@ Apple Comput
9	▬	10910	/usr/dan/users1/dbyrd/wsj89...	Business Brief -- Apple Comp
10	▬	3071	/usr/dan/users1/dbyrd/wsj89...	At Apple Computer@ Proper

Figure 10 (implementation). Combination of query input area (Vquery1) and result list



Apple Stock Jumps on
Earnings

By Catalina Ortiz
AP Business Writer
Thursday, July 16, 1998; 10:45 a.m. EDT

SAN JOSE, Calif. (AP) -- Apple
Computer Inc., showing more signs
of a financial recovery, reported its
third consecutive quarterly profit, a
better-than-expected $101 million.

The news sent Apple stock soaring 6
percent in morning trading, up $2.12
1/2 a share at $36.56 1/4 on the Nasdaq Stock Market.

The popularity of speedy G3 Macintosh computers and a continuing
cost-cutting effort helped swell Apple's earnings for the April-June quarter to
65 cents a diluted share. That compares with a loss of $56 million, or 44 cents
a diluted share, in the same period last year.

Revenues dropped 19 percent, to $1.4 billion from $1.7 billion. But they were
about the same as the previous quarter's, in line with Apple's predictions. And
the company expects to see sales rise with its new iMac computer, which
becomes available next month.

Profits for the spring quarter got a $26 million boost from an investment gain.
Without it, Apple would have earned $75 million, or 50 cents a share.

🟥 apple 🟩 computer 🟦 stock

Figure 11 (implementation). Document viewer with scrollbar visualization (Vquery-doc2)

6. Evaluation

A. Evaluation Completed: Vquery-doc2 with Ordinary Users

Although our motivation for beginning this work involved expert searchers, we are equally interested in "ordinary" users, and Byrd (1999) describes in some detail a formal user study of Vquery-doc2 with a population of students. Here is a summary of the study and its conclusions.

THE EXPERIMENT

We compared an experimental system with visualization to a control system. Note that we were interested only in the document viewer, where the Vquery-doc2 visualization would be used, and the experimental system's document viewer incorporated the full visualization. The control system's viewer had no visualization except for highlighting words in the text in a single color. We made two types of measurements: objective, including comparisons of participants' relevance judgements to the "official" ones, and how quickly they could judge documents; and subjective, i.e., how much they liked using the visualization. To minimize irrelevant differences between the experimental and control systems, the code for the control system's scrollbar was identical to that for the experimental system except that the control system skipped drawing the icons.

Suitable objective measures include the time users need to find specified information, and how much of the available information they find. Measuring actual proportions of information found—i.e., recall—is possible only if one has exhaustive relevance judgements. But now that test collections considered appropriate for IR research are in the gigabyte range and above, exhaustive relevance judgements are hardly available for any collection. However, we needed only to compare how well our participants did with the experimental system to how well they do with the control system; we did this simply by ignoring any unjudged documents users retrieved, effectively assuming that these documents will have roughly the same precision as judged documents.

Participants

All participants were students, adult native speakers of English, from 18 to 30 years old, with at least some experience with computers, and with normal color perception. All had experience with online searching (averaging over three years), but none had professional training or professional experience as a searcher. We ran a pilot study with six participants, four male and two female; the full study had 20 participants.

Tasks

The study was modeled after the TREC 6 Interactive track experiment (Byrd et al 1997). Each participant did the same 10 tasks in the same order; the tasks involved identifying relevant documents in a given database.

Specifically, for each task, we gave the participant a description of an information need, plus—since we were interested only in the document viewer—a fixed query and a fixed number of documents, namely 30, to retrieve. The combination of fixed database, fixed query, and fixed number of documents to retrieve means that, effectively, a result list was predefined for each query. We asked participants to consider each result list and to judge relevance of as many as documents as possible in five minutes.

Database and Topics. For the usual reasons (so we could use TREC relevance judgments, etc.), we used part of the TREC document collection (about 5 GB in all) with information needs from the TREC topic collection (350 topics). These topics run against these documents have extensive, though by no means complete, relevance judgements that are generally believed to be of high quality.

Note that the content-displaying scrollbar is not likely to be of much use with short documents, since a user can browse through such documents very quickly with no more aid than conventional single-color highlighting of query terms. But we wanted to encourage users to rely on our scrollbar icons as much as possible, so we needed long documents. The Federal Register consists of official U.S. government documents. In general, these documents are long; the longest are well over a megabyte. Also, they tend to contain large amounts of "bureaucratese" and/or trivial details, and they have no titles that a program can recognize as such and display, even though most contain something a human being can recognize as a title. All these factors make Federal Register a very difficult place to find information and a potentially fruitful test collection for a document viewer. For this study, we chose the 1989 Federal Register (FR89), one of the TREC Volume 1 document collections.

Queries. Wanting short and unstructured queries, we used TREC topic titles (with minor changes in two cases) as our 10 queries.

Although FR89 contains many long documents, not all queries will find them. We selected queries whose top 30 documents had an average length against FR89 of over 1000 words.

There were several additional criteria for the queries we chose: maximum length of any retrieved document not too high; neither too few nor too many non-stopped terms; top-30 precision neither too high nor too low. For details, see Byrd 1999.

Procedure

First, each participant filled out a questionnaire to give us basic demographic information (age, education, gender, and previous computer experience). Then they took a standard psychometric test, a test of structural visualization. One unexpected and interesting finding of our work in TREC 6 was that structural visualization, a form of spatial ability, appears to be a good predictor of facility with a direct-manipulation interface. Accordingly, we administered the same structural-visualization test we used in TREC 6 (test VZ-2, "paper folding", from ETS; see Ekstrom et al 1976).

Next, the participant was given a tutorial to learn one system, then they worked on the first five topics. After a short break they were switched to the other system, given a tutorial, then worked on the other five topics. Each search had a 5-minute time limit, and the participant was instructed to stop working if they had not finished in 5 minutes.

We gave the participant a short questionnaire after each search. After all the searches were finished we gave them a final questionnaire, then "debriefed" them.

We ran each participant through the entire study in a single essentially continuous period of about two and a half hours. Half did the first five searches with the experimental system, and the other half did the first five with the control system: thus, there were two conditions.

Results

For objective measurements, we analyzed the participant's relevance judgments by comparing them to the official TREC judgments. We then performed an ANOVA (ANalysis Of VAriance) using query, participant, and system as factors. For dependent variables, we used

- Number of documents judged
- Number of documents correctly judged
- Accuracy

In both the pretest and the full study, query- and participant-dependent results were significant. However, we found no system-dependent results. The differences between the experimental system and the control system were what would be expected by chance.

We also made subjective measurements by asking participants whether they preferred the experimental or the control system, and how strong their preference was on a five-point scale ("not at all" to "extremely"). Combining these questions gives nine values. Responses on both the pilot and the full study showed beyond any reasonable doubt a strong preference for the experimental system: in fact, on the full study, the preference for the experimental system was significant at $p < .0001$.

Participants made a number of illuminating comments. Two who preferred the experimental system commented that—while the visualization wasn't always useful—when it was not useful, it did not get in the way. One went on to say that he could not understand why anyone would not prefer the scrollbar icons: "if you want, you can just ignore them."

B. Evaluation Proposed: All Visualizations with Both User Groups

We wish now to evaluate the success of the complete MIRV system with each group—expert searchers and ordinary users — independently. Both objective and subjective measures can be done as in our previous experiment.

Experimental Design

To learn very much from this research, we need to evaluate the contributions of the different visualizations separately, at least to the extent possible. The independent variables we want to consider are:

1. User Group (type of user). The two values are (1) expert searcher; (2) "ordinary" user.
2. Interface for Query Formulation. The three values are (1) experimental system: free text input, with Vquery and Vquery-DB; (2) control 1: free text input with no visualization; (3) control 2: Boolean input with no visualization.
3. Interface for Result List (summary). The two values are (1) experimental system: Vquery-doc1 and show best passage, etc.; (2) control: no visualization and show beginning of document.
4. Interface for Document Viewer. The two values are (1) experimental system: Vquery-doc2 plus highlighting query terms in the text in the same colors as Vquery-doc2; (2) control: no visualization except highlighting words in the text in a single color. (It might also be interesting to study a value "past" the second, i.e., no visualization at all. We will ignore this possibility for now.)

With respect to Interface for Query Formulation, the styles now used by expert searchers are very different from those used by end users. Therefore, we intend to compare our experimental query-formulation system to two different controls: one Boolean system with a syntax similar to, say, Alta Vista advanced search (the type of retrieval system, though not the syntax, that expert searchers generally use), and one best-match system with a syntax similar to, say, Alta Vista basic search (typical of what end users employ on the World Wide Web tens of millions of times per day).

Unfortunately, a straightforward experiment with all four variables would be quite difficult: the full factorial design involves 2 x 3 x 2 x 2 = 24 conditions, and it is not easy to argue that any of the variables are independent so that cells could be omitted. However, an alternative is to do three simpler experiments: User Group against each of the other variables separately. This can be done in the following way:

Experiment A (variables: User Group, and Interface for Query Formulation => 6 conditions). Give the participant several information needs, and ask them to come up with the best query they can for each. Based on experience, 10 is probably the minimum number of information needs that would yield meaningful results, but 15 or 20 would be much better. Evaluate results with standard IR and interactive-IR measures like precision, recall, and top-n precision, after both the first iteration and the last.

Experiment B (variables: User Group, and Interface for Result List => 4 conditions). Give the participant an online list of pre-defined queries—perhaps one for each information need in Experiment A—and ask them to run each query, and to find a fixed number of relevant documents as quickly as possible. Alternatively, ask them to find all relevant documents, and evaluate the results on both the amount of information they locate and on elapsed time.

Experiment C (variables: User Group, and Interface for Document Viewer => 4 conditions). Give the participant an online list of queries and associated result lists. Ask the participant to choose each result list and to find the most relevant passages for the associated query across all documents as quickly as possible. Alternatively, define a number of "subqueries" for each query that are more tightly focused; ask the participant to choose each result list and to find the most relevant passages for each subquery across all documents as quickly as possible. Alternatively, give the participant a number of queries that produce result lists in which nearly all documents appear relevant, but in which many or most are not; ask them to choose each result list and to find a fixed number of relevant documents for the associated query as quickly as possible. For this experiment, the documents should be long ones, since the visualization we are testing is obviously of much less value for short documents.

More specifically, the experiments will look like this. PO = "ordinary participant", PE = "expert participant", T = "task". We assume four participants per group per experiment, and five tasks per cell, though these numbers are subject to change. (For comparison, the TREC-7 Interactive track task specification also calls for participants in groups of four, but eight tasks instead of five.)

Experiment	Test system	Ordinary users	Experts
A. Query Formulation	Free text with Vis	(PO_1...PO_4) x ($T_{1,1}$...$T_{1,5}$)	(PE_1...PE_4) x ($T_{1,1}$...$T_{1,5}$)
	Free text	(PO_1...PO_4) x ($T_{1,6}$...$T_{1,10}$)	(PE_1...PE_4) x ($T_{1,6}$...$T_{1,10}$)
	Boolean	(PO_1...PO_4) x ($T_{1,11}$...$T_{1,15}$)	(PE_1...PE_4) x ($T_{1,11}$...$T_{1,15}$)
B. Result list	With Vis	(PO_5...PO_8) x ($T_{2,1}$...$T_{2,5}$)	(PE_5...PE_8) x ($T_{2,1}$...$T_{2,5}$)
	No Vis	(PO_5...PO_8) x ($T_{2,6}$...$T_{2,10}$)	(PE_5...PE_8) x ($T_{2,6}$...$T_{2,10}$)
C. Document viewer	With Vis	(PO_9...PO_{12}) x ($T_{3,1}$...$T_{3,5}$)	(PE_9...PE_{12}) x ($T_{3,1}$...$T_{3,5}$)
	No Vis	(PO_9...PO_{12}) x ($T_{3,6}$...$T_{3,10}$)	(PE_9...PE_{12}) x ($T_{3,6}$...$T_{3,10}$)

As shown, each participant will be in only one experiment. Thus, we will need a total of 4 x 3 = 12 participants per group, or 24 participants in all. This gives 4 x 5 = 20 data points per cell, enough for a meaningful analysis of variance. Contrary to the simplified presentation shown, to minimize order effects, we will randomize the order in which participants are given the tasks. This might be done either for the five tasks within a cell or for the ten or fifteen in an experiment; the former is simpler and should be adequate.

7. Conclusion

We have described an important problem in text IR; a user-interface-based framework for solving it, with a partial implementation of that framework; a test of part of that implementation; and ways to refine and to further test the implementation.

There are several significant issues that this paper does not address, among them:

- Control of the query-expansion process, e.g., for relevance feedback or LCA.
- Control over long queries (typically resulting from query expansion, or from pasting full text from a document's body). Few if any of our visualizations work well with more than, say, six or seven terms in a query. (An obvious approach to this issue is via clustering query terms, whether manually, automatically, or somewhere in between.)
- Visualization of overlap of query terms in databases as a means of control. This could be done with Venn diagrams, for example. On the other hand, this may not be that significant: it is not clear that simply showing frequencies of overlapping terms without context would help users.
- Handling field-specific aspects of the query.
- Handling database-specific aspects in a multidatabase situation. For example, as we have said, a given raw term may map to different stems in different databases.
- Combining previous queries. This is both well-supported and heavily used in typical Boolean systems.

Even for the issues addressed, we do not claim that the implementation described here is optimal in any way. We present it simply as a starting point for learning more about how to design IR interfaces that allow users of all types maximum satisfaction with best-match searching. But since best-match searching is evidently superior in objective ways to exact-match, this should allow users the greatest satisfaction of any kind of searching.

Acknowledgements

W. Bruce Croft first proposed to one of us (Byrd) re-thinking in terms of visualizations the entire UI for text retrieval; he also gave us much advice on specific visualizations and on experimental design. Leah Larkey gave us detailed advice at several stages of the project, from user interface to experimental design to the content of this paper. Paul Cohen made major contributions to the experimental design. Joanne Claussen of Westlaw and Daniel Pliske of Lexis/Nexis provided information on Boolean versus natural language usage of their respective services. Karen Priore and Andy Wolan administered the user study described in Byrd (1999) and summarized above. Byrd (1998) describes interviews with nine professional librarians, two in public libraries and seven in a large university library; the university librarians were Eric Esau, Larry Feldman, Emily Hurn, Naka Ishii, Pam Juengling, Barbara Morgan, and Emily Silverman. Finally, we are grateful to Pat Billingsley, Jack Conrad, Barbara Morgan, Mark Sanderson, and Susan Schneider for assistance and advice of various sorts.

References

Allan, James (2000). Private communication, April 2000.

Blair, David and Maron, M.E. (1985). An Evaluation of Retrieval Effectiveness for a Full-text Document-Retrieval System. *CACM* 28,3 (March 1985), 289–299.

Byrd, Donald (1998). Conversations with Librarians about Searching. Unpublished manuscript.

Byrd, Donald (1999). A Scrollbar-based Visualization for Document Navigation. *Proceedings of Digital Libraries 99 Conference,* ACM, New York.

Byrd, Donald, Swan, Russell, and Allan, James (1997). TREC-6 Interactive Track Report, Part 1: Experimental Procedure and Initial Results. Tech report IR-117, Computer Science Dept., University of Massachusetts/Amherst.

Claussen, Joanne (1999). Private communication, June 1999.

Croft, W. Bruce, Cook, Robert, and Wilder, Dean (1995). Providing government information on the Internet: Experiences with THOMAS. *Proceedings of Digital Libraries 95 Conference,* ACM, New York. Also available at http://csdl.tamu.edu/DL95/papers/croft/croft.html .

Doan, Khoa, Plaisant, Catherine, and Shneiderman, Ben (1996). Query Previews in Networked Information Systems. *Proceedings of Third Forum on Research and Technology Advances in Digital Libraries, ADL '96,* IEEE CS Press, 120–129. Also available as TR 95-16 at http://www.cs.umd.edu/projects/hcil/Research/tech-report-list.html#1996 .

Ekstrom, R.B., French, J. W., Harman, H. H., and Dermen, D. (1976). *Manual for Kit of Factor-Referenced Cognitive Tests 1976.* Educational Testing Service, Princeton, New Jersey.

Harman, Donna (1992). User-Friendly Systems Instead of User-Friendly Front Ends. *JASIS* 43, 164–174.

Hersh, W.R., and Hickam, D.H. (1995). An evaluation of interactive Boolean and natural language searching with an on-line medical textbook. *JASIS* 46, 478–489.

Hearst, M.A. (1995). Tilebars: Visualization of term distribution information in full text information access. *Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems.*

Hix, Deborah, and Hartson, Rex (1993). *Developing User Interfaces*. New York: John Wiley & Sons.

Pliske, Daniel (1999). Private communication, July 1999.

Rose, Daniel, and Stevens, Curt (1997). V-Twin: A Lightweight Engine for Interactive Use. *Proceedings of TREC-5.*

Shneiderman, Ben (1998). *Designing the User Interface,* 3rd ed. Reading, Massachusetts: Addison-Wesley.

Shneiderman, Ben, Byrd, Donald, and Croft, W.B. (1997). Clarifying Search: A User-Interface Framework for Text Searches. *D-Lib Magazine,* January 1997. Available at http://www.dlib.org .

Shneiderman, Ben, Byrd, Donald, and Croft, W.B. (1998). Sorting out Searching: A User-Interface Framework for Text Searches. *CACM* 41,4 (April 1998), 95–98.

Sparck Jones, Karen (1981). Retrieval system tests 1958–1978. In Karen Sparck Jones, editor, *Information Retrieval Experiment*, 213–255.

Sparck Jones, Karen, and Willett, Peter (1997). *Readings in Information Retrieval.* San Francisco: Morgan Kaufmann.

Tenopir, Carol, and Cahn, Pamela (1994). TARGET and FREESTYLE: DIALOG and Mead join the relevance ranks. *Online* 18,3, 31–47.

Tombros, Anastasios, and Sanderson, Mark (1998). Advantages of Query-Based Summaries in Information Retrieval. *Proceedings of SIGIR '98,* 2–10.

Turtle, Howard (1994). Natural Language vs. Boolean Query Evaluation: A Comparison of Retrieval Performance. *Proceedings of SIGIR '94,* 212–220.